SpaceFOM: An Interoperability Standard for Space Systems Simulations

Edwin Z. Crues,
Daniel E. Dexter
NASA Johnson Space Center
2101 NASA Parkway
Houston, TX 77058
Edwin.Z.Crues@nasa.gov
Daniel.E.Dexter@nasa.gov

Björn Möller
Pitch Technologies
Repslagaregatan 25
S-582 22 Linköping, Sweden
bjorn.moller@pitch.se

Alfredo Garro,
Alberto Falcone
Department of Informatics,
Modeling, Electronics
and Systems Engineering (DIMES)
University of Calabria
via P. Bucci 41C,
87036 Rende (CS) Italy
alfredo.garro@dimes.unical.it,
alberto.falcone@dimes.unical.it

***Abstract*—There is a long history of simulation supporting space systems development. This includes relatively simple parametric simulations to more complex trajectory simulations to large scale integrated vehicle simulation. One area of relatively recent development is in the area of distributed or interoperable simulation. Distributed simulation has been in wide use by the US military for years but is being used more broadly in the aerospace community.**

To support large scale distributed simulation, the military community has developed a number of standards to support a-priori interoperability between large collections of disparate simulations. For example, the IEEE 1516 High Level Architecture (HLA) and the Real-time Platform Reference Federation Object Model (RPR FOM). While HLA is suitable for space systems, there are a number of design decisions made in the development of the RPR FOM that prevent it from working well for space applications.

In order to address these deficiencies, the Simulation Interoperability Standards Organization (SISO) developed a new HLA-based interoperability standard to support the needs of complex space systems. This standard is the Space Reference Federation Object Model (SpaceFOM).

This paper presents an overview of the SpaceFOM including the fundamentals of the SpaceFOM, the key features of the SpaceFOM, and how the SpaceFOM supports large scale distributed simulation of complex space systems.

TABLE OF CONTENTS

978-1-7281-7436-5/21/$31.00 ©2021 IEEE

INTRODUCTION

Now, almost 50 years after the success of the Apollo program, the United States' National Aeronautics and Space Administration (NASA) is actively planning to return humans to the surface of the Moon with the Artemis program[1]. As with the Apollo program, every aspect of the Artemis missions will be modeled and simulated. Fortunately, there have been numerous advances in computational, modeling and simulation technologies over the past 50 years. One of the most notable advances is in the area of collaborative or distributed simulation.

With the advent of large scale, high bandwidth, low latency, and commonly available computer networking using standardized protocols, the distribution of computational problems across connected computers now provides a powerful computational model for large scale collaborative or distributed simulations. One of the leaders in distributed simulation is the United States Department of Defense (DoD). which has been actively developing distributed simulation technologies and distributed simulations for many years. For example, the IEEE 1516 High Level Architecture (HLA) [1] [2] [3] and the Real-time Platform Reference Federation Object Model (RPR FOM) [4]. The military uses standards like these to create large scale distributed simulations involving thousands of computational entities on hundreds of computer systems modeling high complex interactive systems. Most of these systems are limited to terrestrial or Earth-centric problem domains.

In recent years, the broader aerospace community has been adopting distributed simulation technologies to model civil and commercial systems. For instance, NASA is using distributed simulation technologies to support visiting vehicle analysis and training for the International Space Station (ISS) [5] [6]. However, early experiences with existing distributed simulation standards lead to the realization that, while the military standards are very effective for Earth-centric modeling, the standards have limitations that make modeling space systems difficult if not impossible.

For instance, the RPR FOM uses a single Earth-fixed coordinate system. While this works for systems on or near the Earth, it breaks down rapidly much beyond the altitude of geostationary satellites, does not work well for the Moon, and is completely impractical for Mars. Also, many space systems simulations require stringent synchronization of their execution cycles to coordinate dynamic state responses. While HLA supports this, many of the other distributed simulation standards do not. Finally, these systems do not specify details of initialization and execution control that are necessary for a-priori interoperability.

In order to address these and other deficiencies, the Simulation Interoperability Standards Organization (SISO) developed a new HLA-compatible interoperability standard to support the needs of complex space systems. This standard is the Space Reference Federation Object Model (SpaceFOM) [7] [8] [9] [10].

This paper presents a high-level description of the SpaceFOM. This includes a general overview of the SpaceFOM along with more detailed descriptions of key features including: roles and responsibilities; rules and guidelines; documentation; the constituent Federation Object Models (FOMs); a-priori interoperability, robustness, and extensibility attributes; and some design patterns used. This paper ends with a few example use cases for the SpaceFOM.

OVERVIEW

The SpaceFOM standard delineates a prescriptive collection of policies, processes, documented agreements, and HLA constructs intended to provide a sound basis for a-priori HLA-based interoperability for collaborative distributed simulations in the space domain [9] [11].

The SpaceFOM has been defined to meet the following main requirements for supporting the distributed simulation of space missions:

- handling of specific roles and responsibilities of federates within a federation execution;
- management of common data types useful in the space domain;
- management of common time lines and time scales needed for time homogeneity;
- dealing with specific time-stepped focused time management approaches;
- handling of a flexible positioning system, using reference frames for locating arbitrary bodies in space;
- adopting of a naming convention for operational reference frames;
- offering base object support for physical entities (e.g., space vehicles and astronauts);
- offering base object support for physical interfaces (e.g., docking ports and sensor locations);
- handling a synchronized execution control strategy and framework;

[1] In the religion of the ancient Greeks, Artemis is a goddess and the twin sister of Apollo.

- providing rules for assessing the compliance with the SpaceFOM;
- providing a core base set of FOM modules needed for a SpaceFOM-compliant federation execution.

The SpaceFOM identifies specific Federation Execution management roles, a collection of compliance rules, two document templates, and a set of base HLA data constructs contained in a collection of Federation Object Model modules (FOM modules) [1] [2] [3]. The management roles define principal responsibilities in coordinating a Federation Execution and providing critical data during initialization and run-time. The rules codify fundamental actions, relationships, and behaviors required for functional interoperability. The document templates provide an outline for specifying a Federation wide agreement on the fundamental aspects defining a specific Federation Execution and an outline for a document that each Federate must provide defining their level of SpaceFOM compliance. The SpaceFOM FOM modules define a collection of base data types, synchronization points, hierarchical *ObjectClass* definitions and *InteractionClass* definitions that are organized according to their purposes in separate modules (files). This separation provides developers with a flexible and effective means for managing and extending the standard

ROLES AND RESPONSIBILITIES

While the concept of the Federation Object Model (FOM) is contained in the name SpaceFOM, HLA-based interoperability requires more than the data elements contained in a collection of HLA-based FOM modules. Specifically, in addition to the associated FOM modules, the SpaceFOM defines principal roles for a SpaceFOM-compliant Federation and these roles have specified responsibilities. The SpaceFOM defines three principal Federate roles [9]:

Master - Responsible for high level coordination of any SpaceFOM-compliant Federation Execution. The Master Federate supports the Federate role determination process, coordinates the Federation Execution initialization process, and manages the execution moding process.

Pacing - Responsible for coordinating time management and synchronization of a Federation Execution. The Pacing Federate determines the rate at which HLA logical time progresses with respect to this Federate's computer clock. In some cases, this clock may be linked to Central Timing Equipment (CTE) for hardware level synchronization between physically independent Federates and possibly avionics systems.

Root Reference Frame Publisher (RRFP) - Responsible for publishing the name of the root reference frame of the reference frame tree for the Federation Execution. The RRFP Federate provides the name of the root reference frame for the current Federation Execution. This forms the common base or root of the Federation Execution's reference frame tree.

Note that these roles are not mutually exclusive and can coexist within a Federate.

RULES AND GUIDELINES

In addition to the roles and responsibilities defined above, the SpaceFOM standard specifies 103 compliance rules and a few associated guidelines to facilitate a-priori interoperability. These rules cover topics associated with general HLA compliance, documentation, time management, reference frame specification, data specification, and execution control.

The format for a rule is Rule Title, Requirement Statement, and Rationale Statement. For example:

Rule <C>-<#>: Brief Cross-Referenced Numbered Rule Title

Requirement: A rule shall have a brief numbered rule title, a requirement "shall" statement, and a rationale statement. The rule title number is a two part hyphenated number with the first number <C> corresponding to the chapter number and the second number corresponding to the ordered occurrence of that rule within the chapter.

Rationale: This rule format permits the unique specification of a SpaceFOM rule. The **Title** provides a brief descriptive phrase appropriate for listing in the prefatory section of the document. The *Requirement* provides a statement of the rule using "shall" based requirements language. This requirement is the authoritative statement for the rule. The *Rationale* provides context and explanation of the rule as it pertains to the SpaceFOM.

Many sections of the SpaceFOM also provide guidance on the intention of the authors and some examples for clarification.

DOCUMENTATION

The SpaceFOM standard also provides two document templates to assist Federation construction and integration: the *Federation Execution Specific Federation Agreement (FESFA)* and the *Federation Compliance Document (FCD)* [7] [12]. Several rules in the SpaceFOM put requirements on what data needs to be recorded in the FESFA and what data needs to be recorded in the FCDs.

FESFA

The FESFA is a document that represents a Federation-wide agreement between participating Federates and pertains to a specific common set of Federation Executions. The FESFA provides the general purpose and description of a specific SpaceFOM-compliant federation execution. This should include intended scenarios and other information that describes the nature of the federates participating in a federation execution compliant with this FESFA. A FESFA template is provided in the appendix of the SpaceFOM standard document.

A SpaceFOM-compliant FESFA contains eight major sections: Purpose, Identification, Federation Composition, Time management, Reference Frames, Object Management, Initialization, and Additional Technical Information. The Purpose provides a high-level brief description of the purpose, capabilities, and intended use of an associated Federation Execution. The Identification section provides general identifying information associates with the federation execution; this includes federation title, execution name, points of contact, and planned execution dates. The Federation Composition section provides the identifying information associates with the composition of the federation execution; this includes the name(s) of the Master, Pacing, RRFP, and any other required federates. The Time Management section provides the general time management information associates with the federation execution. All participating time managed federates will need this information. Some of the Time Management information is published by the Master Federate in the *Execution Control Object (ExCO)*. The Reference Frames section provides the names and descriptions of the principal reference frames published during a federation execution. It should be sufficient to understand the principal topology of the federation execution's reference frame tree. The Object Management section provides the general object management information associated with the federation execution; this includes the principal FOM modules used, *PhysicalEntity* object type string, *PhysicalEntity* object status strings, and *PhysicalInterface* object instance naming convention. The Initialization section provides the information associates with the initialization policy and approach use in the federation execution. It specifically focuses on the details of any multiphase initialization. All Early Joiner federates participating in the multiphase initialization process will need this information. Finally, the Additional Technical Information section provides any additional technical information needed by federates participating in the federation execution. This section may be marked (N/A) or omitted if there is no additional technical information.

FCD

In contrast to the FESFA, which is a cross-federation agreement, the FCD describes the capabilities of a specific Federate and which roles it can play in a SpaceFOM-compliant Federation Execution. The FCD provides the general purpose and description of a specific SpaceFOM-compliant federate. This should include intended scenarios and other information that describes the nature of the federate's capabilities and compliance as a SpaceFOM-compliant federate. Federate providers should provide a federate compliance declaration to facilitate the assessment of the suitability of a federate in a specific federation execution. An FCD template is provided in the appendix of the SpaceFOM standard document.

A SpaceFOM-compliant FCD contains nine major sections: Purpose, Identification, SpaceFOM Federate Roles Supported, Time management, Reference Frames, Object Management, Initialization, Additional Technical Information, and Compliance Statement [7]. The Purpose section provide the general purpose and description of this specific SpaceFOM-compliant federate. This should include intended scenarios and other information that describes the nature of the federate's capabilities and compliance as a SpaceFOM-compliant federate. The Identification section provides the general identifying information associated with this federate; this includes the federate name, version, and points of contact. The SpaceFOM Federate Roles Supported section provides information on the SpaceFOM roles that this federate can support. The Time Management section provides the general time management information associates with this federate. The Reference Frames section provides the names and descriptions of the principal reference frames published by or required by this federate. The Object Management section provides the general object management information associated with the federate; this includes the principal FOM modules provided or needed, *PhysicalEntity* object type strings used, *PhysicalEntity* object status strings used, and the names of *PhysicalInterface* object instances used. The Initialization section provides the information associates with the initialization policy and approach use by this federate. It specifically focuses on the details of its M*ultiphase Initialization Process (MPI)*. The information in this section will inform the overall MPI strategy for a federation execution in which this federate participates. The Additional Technical Information section provides any additional technical information needed by this federate. Finally, the Compliance Statement section provides a general acknowledgement that this federate complies with the Space Reference FOM standard.

FOM MODULES

Of course, the SpaceFOM also defines the base set of HLA-compliant FOM modules. Figure 1 shows the five FOM modules that constitute the SpaceFOM along with the architecture and module dependencies. These modules are: *SISO_SpaceFOM_switches*, *SISO_SpaceFOM_*datatypes, *SISO_SpaceFOM_environment*, *SISO_SpaceFOM_management*, and *SISO_SpaceFOM_entity* [7] [9].

The SpaceFOM modules, as all HLA FOMs, relies on the *Management and Initialization Module (MIM)* that contains the *Object Model Template (OMT)* tables that describe the *Management Object Model (MOM)*, which is used to control and monitor a federation execution

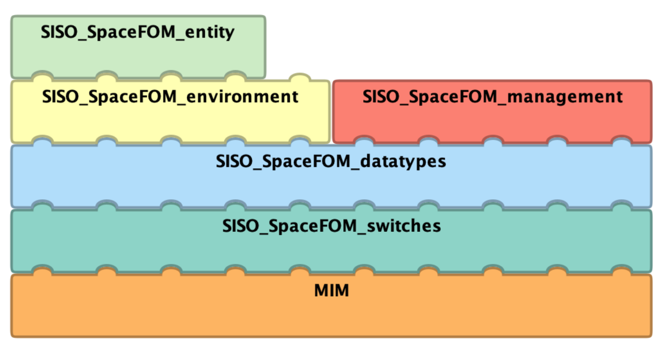


Figure 1: Architecture of the SISO SpaceFOM.

***SISO_SpaceFOM_switche*s** – This module provides configuration settings for the Federation execution by way of global Federation execution wide switches for Local Run-Time Component (LRC) and RTI behavior. The IEEE 1516-2010 standard defines a set of switches that shall be set in the FOM. These switches regulate the behavior of some of the optional actions the RTI can perform on behalf of the Federate, such as automatically requesting updates of an instance attribute when an object instance is discovered or advising the Federates when certain events occur. To facilitate easy replacement of these settings, the switches have been confined to the SISO SpaceFOM switches FOM module. It is expected that Federations might choose to update this module based on their Federation agreement.

SISO_SpaceFOM_datatypes - This module provides the definitions of fundamental data types used as a basis for commonality between SpaceFOM-compliant Federates. This includes three principal HLA data types:

simpleDataTypes - contains representations for the main scalar physical quantities, such as Angle, Mass, MassRate, Velocity and Acceleration;

arrayDataTypes - includes the definitions for managing vector physical quantities, such as position, velocity and acceleration;

fixedrecordDataTypes - contains representations for the space-time coordinates and reference frame states.

This FOM module also defines the HLA logical timestamp and lookahead time; both are represented as 64 bits integers, *HLAinteger64Time*. These data types are used for object attributes as well as interaction parameters and adopt the International System of Units (SI) wherever possible. In addition, this module defines the *SpaceTimeCoordinate ObjectClass* that provides the base information for representing when and where any reference frame or physical entity exists in time and space.

SISO_SpaceFOM_environment - This module provides the fundamental data types used to represent the basic physical environmental properties associated with space-based simulations. In particular, it defines the *ReferenceFrame* HLA *ObjectClass* that provides the base information for associating reference frames and forms the basis for coordinate and state transformations.

SISO_SpaceFOM_management - This module offers the specifications for execution control and management of HLA *ObjectClass*, *InteractionClass* and *SynchronizationPoint* instances. Specifically, it defines the base set of information necessary to coordinate Federation and Federate execution time lines and execution mode transitions in a SpaceFOM compliant Federation Execution.

SISO_SpaceFOM_entity - This module provides the basic state definitions of any physical object in a space environment through the definition of the *PhysicalEntity*, *DynamicalEntity*, and *PhysicalInterface ObjectClasses*. A *PhysicalEntity* is the fundamental base class that provide state information for any item physically present in the Federation Execution. A *DynamicalEntity* inherits from *PhysicalEntity* and can be used to represent a man-made vehicle or a major sub-element of a man-made vehicle. A *PhysicalInterface* is used to create geometric associations between a *PhysicalEntity* or another *PhysicalInterface*.

A-PRIORI INTEROPERABILITY, ROBUSTNESS, AND EXTENSIBILITY

Fundamentally, the purpose of the SpaceFOM standard is to provide a codified process for creating HLA-based Federates that can reasonably be expected to work together without significant additional negotiation and integration. This is the concept of a-priori interoperability for space systems simulations [10] that is a key enabling factor for the Distributed Digital Twin (DDT) paradigm. In addition to this, the SpaceFOM is intended to provide for robust execution of the constituent Federates and provide for extension through a common set of base capabilities and data types. Robustness and extensibility are two important aspects of the SpaceFOM standard since, on one hand, it has been defined to be robust even in the presence of unexpected inputs and behavior of the simulation models, and on the other hand, it provides mechanisms to extend the offered functionalities to meet specific requirements of a specific space mission or even a campaign of space missions. Extensions can be through the definition of new functionalities or through modification of existing ones without impairing the existing functions, roles, and constraints. Finally, through its robustness and making use of base extension, the SpaceFOM offers mechanisms to detect failing Federate and Federation Executions, allowing the operator to take action.

The next section reports a set of design patterns, introduced during the SpaceFOM definition process, to enable or contribute directly to the extensibility, interoperability, and robustness of the standard.

USING DESIGN PATTERNS

In software engineering, design patterns are general repeatable solutions to common problems in software design. A pattern does not represent a finished design that can be translated directly into code, but is a template for addressing a specific design problem [11].

During the SpaceFOM standardization process, several domain-independent and domain-specific design patterns have been introduced to deal with design, development, coordination, and execution challenges of complex systems [9] [11] [13]. Although they have been conceived with reference to the typical issues of the distributed simulation of space missions and systems, it is worth noting that the applicability of the introduced design patters (especially the domain-independent ones) can be exploitable as reference solutions for addressing general distribute simulation issues (e.g., synchronization, coordination and time management) in different application domains.

The design patterns used in the SpaceFOM are typically associated with the specific SpaceFOM roles of *Master*, *Pacing*, and *Root Reference Frame Publisher Federates* and can be segregated into the functional areas of *Execution Control, Time Management,* and *Spacial Definition*. For instance, the *Master* role controls initialization and execution of the Federation; the *Pacing* role managed the advancement of scenario time in relationship to real-time; and the *Root Reference Frame Publisher* role defines the foundational reference frame for a Federation Execution's reference frame tree.

Many of the SpaceFOM patterns rely on both HLA synchronization points and HLA time management services, which make it possible for Federates to manage simulation time, and pause and wait for all Federates to complete their processing and proceed to the next step in a fully synchronized way [9].

Execution Control

This section presents six design patterns used to enable the initialization and execution control processes of the Federates in an associated Federation Execution. These patterns are:

- Federation Execution orphan detection, creation, and join;
- Centralized checking for required federates;
- Detection and designation of early and late joining federates;
- Global configuration using a singleton instance;
- Synchronized multi-phase initialization;
- Central execution control with transition requests.

Time Management

This section presents the design patterns for handling four time concepts delineated in the SpaceFOM standard: *Simulation Scenario Time, HLA Logical Time, Computer Clock Time,* and *Physical Time*. *Simulation Scenario Time (SST)* is the conceptual time associated with the modeled systems. *HLA Logical Time (HLT)* represents the time used by HLA to timestamp messages, order messages, and regulate time advance. This time concept is related to SST through a starting point or epoch (SST0); usually, HLT starts at zero. The *Physical Time* is based on the classical Newtonian concept of absolute real-world time. *Computer Clock Time (CCT)* is the model for time used by the computer to represent Physical Time. Three time management patterns, closely related to these time lines and SpaceFOM execution control, have been defined:

- Constant but potentially different federate time steps;
- Coordinated execution time lines and pacing;
- Distributed hardware-based real-time pacing.

Reference Frame Management

Space simulations are composed of models that are often formulated with respect to specific reference frames; they are abstract coordinate systems that allow, through a set of reference points, to locate and orient physical objects in space and time.

In any Federation Execution, one Federate may have a preferred computationally convenient reference frame, whereas another Federate may use a different one. So, how does one Federate work with the data from another Federate if they have different representational frames? The answer is that every SpaceFOM-compliant Federation Execution has a rooted directed acyclic graph of reference frame associations, also known as a *Reference Frame Tree* that provides transformations between Federate reference frames through these two design patterns:

- Reference frames explicitly specified using object instances;
- Replaceable and extendable tree of reference frames.

SPACEFOM IN USE

While the SpaceFOM is a relatively new standard for space systems simulation interoperability, it already has some examples of successful use in application to space systems simulation. Three use cases will be discussion in this section: the Simulation Exploration Experience (SEE), the Harwell Robotic and Autonomy Facility (HRAF), and the NASA Artemis program.

Simulation Exploration Experience (SEE)

The *Simulation Exploration Experience (SEE)* project, organized since 2011 by SISO in collaboration with NASA and other industry and research partners, aims at providing to undergraduate and postgraduate students a practical experience on Distributed Simulation (DS) systems compliant with the IEEE 1516-2010 standard [14] [15] [16].

The SpaceFOM version 0.1 was successfully experimented during the 2017 edition of the SEE project. In this edition, in which eleven universities were involved (University of Alberta, University of Nebraska-Lincon, the Faculdade de Engenharia de Sorocaba FACENS, University of Calabria, University of Genoa, University of Bordeaux, University of Munich, University of Brunel, University of Liverpool, University of Jaipur, and University of Bulgaria), a lunar settlement was simulated with a dangerous scenario involving an asteroid on a collision course with the Moon.

The SpaceFOM has been enhanced on the basis of the results of the SEE 2017 edition in order to increase the stability and dependability of compliant Federates. The updated version 0.2 of the SpaceFOM was then experimented in the 2018 edition of the SEE project, where ten universities participated to simulate a settlement on both Moon and Mars. All of the teams developed a 3D model for their Federates to interact with Distributed Observer Network (DON), a real-time 3D visualization environment developed by NASA that allows to track the Federates' activities and displays updates on the 3D environment during the simulation execution through the DON Visualization Tool (DON-VT). The SEE HLA Starter Kit was utilized by the majority of SEE teams to create their Federates compliant with the SpaceFOM; indeed, it provides a full-fledged Java-based development framework that eases the implementation of Federates complaint with both the HLA and SpaceFOM standard [17].

Due to the geographically distributed nature of the SEE project, it represents an important test-case for the SpaceFOM: SEE partners use heterogenous technologies, operate concurrently on the same simulation scenario and infrastructure, and interact mainly through remote synchronous and asynchronous collaborative tools. This is a typical case where the a-priori interoperability plays a crucial role. As well as testing the SpaceFOM standard, the concrete and continuative exploitation of it in the SEE project provides useful insights and indications on how to evolve the standard itself, paving the way for future releases.

ESA – HRAF3

The Harwell Robotic and Autonomy Facility (HRAF) activities funded by the European Space Agency (ESA) aim to provide advanced capabilities to support the development and testing of complex autonomous systems for the exploration of our Solar System. The outcome of one of these activities is a flexible simulation environment allowing models and real hardware to be combined, compared and tested in a plug and play mode [18].

One scenario is concerned with Mars Sample Return (MSR). Specifically, the mission phase where the Orbiting Sample (OS) is retrieved by a Chaser spacecraft (ERO) in Mars orbit for later return and analysis on Earth. The guidance, navigation and control (GNC) functionality using Image based Navigation techniques is accompanied by a high-fidelity Physics "real-world" simulator. Another scenario is concerned with the soft, precision landing of a Spacecraft on a low gravity Near Earth Object (NEO).

The federation is based on the High Level Architecture [1], together with the associated SISO Space Reference FOM standard [7]. Different configurations of the Federation are constructed, the MSR Scenario considering a Model-in-the-Loop (MIL) and Hardware-in-the-Loop (HIL) configuration and the NEO Scenario implementing MIL, Processor-in-the-Loop (PIL) including Synthetic Image generation and HIL configurations. In many cases the same functionality is provided as MIL, PIL and HIL and federates can be exchanged between executions. The federation can also be run locally or distributed between ESA and contractor sites.

NASA – NExSyS and Artemis

At NASA's Johnson Space Center, the NASA Exploration Systems Simulations (NExSyS) team employed some of NASA's principal modelling and simulation tools to explore, develop, and test the SpaceFOM. Two of NASA's principal simulation development tools are the Trick Simulation Environment [19] and the Trick High Level Architecture interface package [20]. Many simulations have been developed in Trick to support NASA's human exploration missions. TrickHLA is a Trick-compatible interface package that provides HLA-based interoperability with these simulations. TrickHLA was developed prior to the specification of the SpaceFOM standard. As part of the SpaceFOM development process, TrickHLA was extended to support full SpaceFOM compliance by adding new SpaceFOM functionalities (i.e., role responsibilities, initialization sequencing, time standards, reference frame publication, and execution control).

The NExSyS team is using these tools, HLA, and the SpaceFOM to support simulation development for NASA's Artemis Program [21]. The NExSyS team is building a distributed simulation architecture based on HLA and SpaceFOM to support the formulation and development of the Artemis Base Camp. Figure 2 shows a conceptual Artemis Base Camp with key lunar exploration elements.

Figure 2: Artemis Base Camp

The NExSyS team is working with NASA domain experts, industry partners, and international space agencies to model prospective Artemis elements and link them together in a SpaceFOM-compliant HLA-based federation execution.

CONCLUSIONS

Space exploration is already a collaborative activity and will only become more so in the future. Unfortunately, with collaboration comes additional concerns and constraints. Coordinating interfaces becomes a challenge, further complicated if proprietary information is involved. Distributed simulation is a proven approach to dealing with some of those challenges and constraints but requires a coordinated computational infrastructure.

For decades now, HLA has been widely used by the military to provide a base interoperability infrastructure. While HLA does provide a sound computational base for distributed simulation in general, it does not address some necessary aspects required for a-priori interoperability in space systems distributed simulations. The SpaceFOM was developed to provide the additional infrastructure to support a-priori interoperability for space systems distributed simulations. The SpaceFOM identifies specific Federation Execution management roles, execution control processes, a collection of compliance rules, two document templates, and a set of base HLA data constructs contained in a collection of FOM modules.

This paper presents the fundamental aspects of the SpaceFOM along with references for more detailed exploration and some examples of the SpaceFOM in use. While the SpaceFOM is a relatively new standard, it is already in active use by notable organizations in the space domain. The authors, also in the context of the Distributed Digital Twin (DDT) paradigm they are delineating, encourage the reader to further explore the SpaceFOM, participate in its development, and share their experiences and ideas for extension.

ACKNOWLEDGEMENTS

The authors would like to acknowledge and thank a number of organizations and people. We would like to thank the Simulation Interoperability Standards Organization for providing the organizational support and standardization framework for developing a broad collection of simulation interoperability standards in general and the SpaceFOM standard specifically. We would like to thank NASA for permitting NASA employees to participate in the development, testing, drafting, and ongoing support of the SpaceFOM standard. We would specifically like to thank Doug Craig (head of the Human Exploration Office, Systems Engineering and Integration, Strategy and Architecture Team) for the funding, faith, advice, and direction. Without this support, the SpaceFOM would not exist.

BIBLIOGRAPHY

[1] IEEE, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework Rules, Vols. IEEE 1516-2010, New York, New York, New York: IEEE, 2010.

[2] IEEE, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification, Vols. IEEE 1516.1-2010, New York, New York: IEEE, 2010.

[3] IEEE, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Object Model Template (OMT) Specification, Vols. IEEE 1516.2-2010, New York, New York: IEEE, 2010.

[4] Simulation Interoperability Standards Organization, Standard for Real-time Platform Reference Federation Object Model (RPR FOM), Orlando, Florida, USA: Simulation Interoperability Standards Organization, 2015.

[5] G. Lauderdale, E. Z. Crues, D. Snyder and D. Hasan, "A Feasibility Study for ISS and HTV Distributed Simulation," in *AIAA Modeling and Simulation Technologies Conference and Exhibit*, Austin, Texas, 2003.

[6] G. Lauderdal, E. Z. Crues, D. Snyder and D. Hasan, "Further Studies On The Feasibility Of A Distributed ISS and HTV Simulation," in *Fall 2003 Simulation Interoperability Workshop and Conference*, Orlando, Florida, 2003.

[7] Simulation Interoperability Standards Organization, Standard for Space Reference Federation Object Model (SpaceFOM), Orlando, Florida, USA: Simulation Interoperability Standards Organization, 2020.

[8] B. Möller, E. Z. Crues, D. E. Dexter, A. Garro, A. Skuratovsky and A. Vankov, "A First Look at the Upcoming SISO Space Reference FOM," in *SISO 2016 Simulation Innovation Workshop (SIW)*, Orlando, Florida, 2016.

[9] E. Z. Crues, D. Dexter, A. Falcone, A. Garro and B. Möller, "SpaceFOM - A Robust Standard for Enabling A-Priori Interoperability of HLA-Based Space Systems Simulations," *Journal of Simulation,* no. DOI:10.1080/17477778.2021.1945962, pp. 1-21, 2021.

[10] B. Möller, A. Garro, A. Falcone, E. Z. Crues and D. Dexter, "Promoting a-priori interoperability of hla-based simulations in the space domain: The SISO space reference FOM initiative.," in *20th IEEE/ACM international symposium on distributed simulation and real time applications, DS-RT 2016*, London, United Kingdom, 2016.

[11] B. Möller, E. Z. Crues, D. Dexter, A. Garro, M. Madden and A. Skuratovsky, Design and Principles Enabling the Space Reference FOM, Orlando, Florida: SISO, 2018.

[12] B. Möller, "Developing Space Reference FOM Federation Agreements," Orlando, Florida, USA, 2019.

[13] B. Möller, A. Garro, A. Falcone, E. Z. Crues and D. Dexter, "On the execution control of HLA federations using the SISO space reference FOM," in *21st IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications, DS-RT 2017*, Rome, Italy, 2017.

[14] A. Falcone and A. Garro, "Using the HLA Standard in the context of an international simulation project: The experience of the 'SMASHTeam'," in *15th international conference on Modeling and Applied Simulation, MAS 2016*, 2016.

[15] A. Garro, A. Falcone, N. R. Chaundry, O. Salah, A. Anagnostou and S. J. Taylor, "A prototype HLA development kit: results from the 2015 Simulation Exploration Experience," in *3rd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, ACM SIGSIM PADS 2015*, 2015.

[16] S. J. Taylor, J. Reyagar, J. Chamber, M. Yero, A. Anagnostou, A. Nouman, N. R. Chaudhry and P. R. Elfrey, "Simulation exploration experience: A distributed hybrid simulation of a lunar mining operation," in *2014 IEEE/ACM 18th International Symposium on Distributed Simulation and Real Time Applications, DS-RT 2014*.

[17] A. Falcone, A. Garro, S. J. Taylor, A. Anagnostou, N. R. Chaudhry and O. Salah, "Experiences in simplifying distributed simulation: The HLA development kit framework," *Journal of Simulation,*

vol. 11(3), no. DOI 10.1057/s41273-016-0039-4, pp. 208-207, 2017.

[18] B. Möller, T. Gray, S. Kay, A. Kisdi, K. Buckley and J. Delfa, "Experiences from the SISO SpaceFOM at the European Space Agency," in *SISO 2021 Simulation Innovation Workshop*, Orlando, 2021.

[19] NASA, "Trick Simulation Development Environment," 31 August 2021. [Online]. Available: https://nasa.github.io/trick/. [Accessed 1 September 2021].

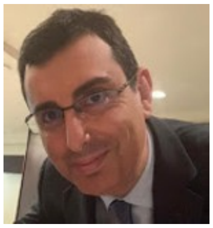
[20] NASA, "TrickHLA - Trick High-Level Architecture (HLA) Framework for Facilitating IEEE 1516 Simulation Integration," 31 August 2021. [Online]. Available: https://github.com/nasa/TrickHLA/wiki. [Accessed 1 September 2021].

[21] National Aeronautics and Space Adminstration, "Artemis Plan: NASA's Lunar Exploration Program Overview," NASA, Washington, D.C., 2020.

BIOGRAPHIES

Edwin Z. {Zack} Crues *received B.S., M.S., and Ph.D. degrees in Aerospace Engineering from the University of Texas, Austin in 1983, 1985, and 1989 respectively. Zack has over 30 years of professional experience in developing spacecraft simulation and simulation technologies. Zack is currently a member of the Simulation and Graphics branch at NASA's Johnson Space Center in Houston, Texas where he leads the development of simulation technologies and the application of those technologies in the simulation of NASAs current and proposed crewed spacecraft. He has developed hundreds of models and simulations for NASA spacecraft, European Space Agency launch systems, and Japanese Aerospace Exploration Agency spacecraft. Zack's recent research focus has been developing and applying distributed computation and distributed simulation technologies. This includes a large-scale distributed simulation of NASAs proposed human space exploration missions. Zack has received numerous awards from the Simulation Interoperability Standards Organization (SISO) including the SISO Technical Merit Award and from NASA including JSC Director's Commendation Awards and the NASA Exceptional Engineering Achievement Medal. His email address is edwin.z.crues@nasa.gov.*

Björn Möller *is the President and co-founder of Pitch Technologies. He leads the development of Pitch's products. He has more than twenty-five years of experience in international high-tech R&D companies, in areas such as modeling and simulation, artificial intelligence and web-based collaboration. Björn Möller holds a M.Sc. in Computer Science and Technology after studies at Linköping University, Sweden, and Imperial College, London. He currently serves as the chair of the SISO Space FOM Product Development group, the chair of the SISO RPR FOM Product Development Group. and the vice chair of the SISO HLA Evolved Product Development Group. His email address is bjorn.moller@pitch.se.*



Alfredo Garro *received a PhD in Systems and Computer Engineering from the University of Calabria (Italy), where he is currently an Associate Professor of Computer and Systems Engineering with the DIMES. In 2016, he was Visiting Professor at NASA JSC, working with the Software, Robotics, and Simulation Division (ER). His main research interests include M&S, multi-agent systems, systems and software engineering, and reliability engineering. His list of publications contains more than 100 papers published in international journals, books, and proceedings of international and national conferences. He is the co-founder (in 2014) and director (from 2018) of the Departmental Research Laboratory "System Modeling And Simulation Hub Lab (SMASH Lab)". He is vice chair of the SpaceFOM PDG and PSG of SISO. He is the President of the ''Italian Chapter'' of INCOSE. He is a member of the Executive Committee of the SEE (Simulation Exploration Experience) project. He is involved as a IEEE Senior Member in the activities of the IEEE Computer Society, IEEE Reliability Society, and IEEE Aerospace and Electronic Systems Society. His email address is alfredo.garro@dimes.unical.it.*

Alberto Falcone *is a Postdoc Research Fellow at the Department of Informatics, Modeling, Electronics and Systems Engineering (DIMES) of the University of Calabria (Italy). From January to October 2016 he was a Visiting Scholar/Researcher at NASA JSC, working with the Software, Robotics, and Simulation Division (ER). His main research interests include M&S, distributed simulation, systems and software engineering. He is a member of the Simulation Interoperability Standards Organization (SISO). He is a member of the SpaceFOM PDG and PSG of SISO. He is a Member of the Executive Committee of the SEE project as Technical Advisor. He was Tutorial Chair of the 2019 Spring Simulation Conference (SpringSim'19). His email address is alberto.falcone@dimes.unical.it.*

Dan Dexter *is an engineer in the Simulation & Graphics Branch in the Software, Robotics and Simulation Division of the Engineering Directorate at NASA's Johnson Space Center in Houston, Texas. He has over 22 years of software and simulation development experience ranging from nonlinear signal and image processing, distributed supercomputing, and flight related software to national and international distributed simulations. He is the principal developer of the TrickHLA software package, a NASA developed middleware software package for using the HLA distributed simulation standard with NASA standard M&S tools.*